**LA HERRAMIENTA MAVEN**

La mayor utilidad de Maven consiste en que esta herramienta es un todo en uno, ya que sus funcionalidades abarcan varios campos tales como el compilado, testear, empaquetar y la gestión de versiones.

**FASES**

Las más relevantes son:

**validate**: Valida que el proyecto es correcto y que está disponible toda la información necesaria

**compile**: Compila el código fuente del proyecto

**test**: Prueba el código compilado utilizando algún framework de pruebas unitarias.

**package**: Coge el código compilado y lo empaqueta en un formato distribuible. Por ejemplo, como un JAR.

**verify**: Ejecuta comprobaciones de pruebas de integración para asegurarse que se cumplen los criterios de calidad

**install**: Instala el paquete en el repositorio local (el directorio ${user.home}/.m2) para que se pueda usar como dependencia en otros proyectos localmente

**deploy**: Copia el paquete final a un repositorio remoto para compartirlo con otros desarrolladores. Hay que tener en cuenta que este despliegue no se refiere en general al despliegue de la aplicación en un servidor web, sino a dejarlo disponible en un repositorio para que lo usen otros desarrolladores.

**CICLOS**

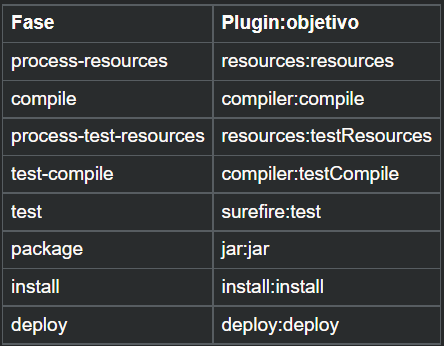
El ciclo de vida default, que gestiona la construcción y despliegue del proyecto.

El ciclo de vida site, que gestiona la creación de la documentación del proyecto.

El ciclo de vida clean, que gestiona la limpieza del directorio del proyecto. Es decir, se encarga de eliminar todos los archivos generados en el proceso de construcción y despliegue.

**PLUGINS**

Los plugins dependen de la fase, la siguiente es una tabla con los plugins acompañados de sus fases.



**¿Para qué sirve el repositorio central de Maven?**

Al igual que GitHub, es un repositorio que permite el versionamiento

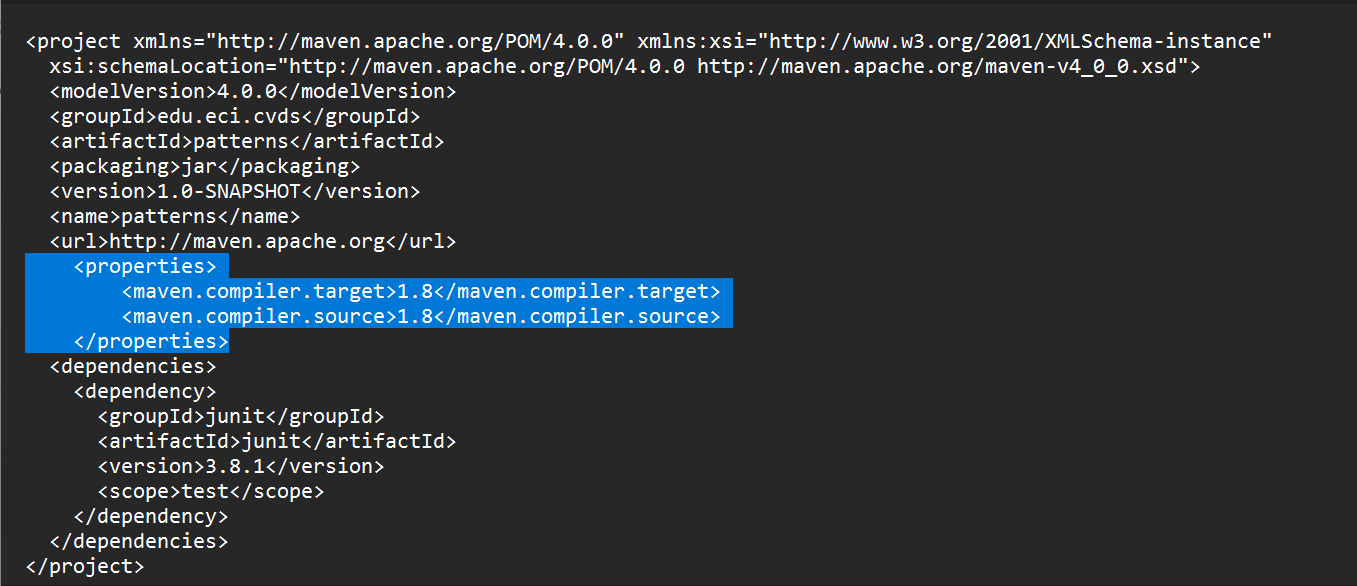
**CREAR UN PROYECTO CON MAVEN**

Comando usado: mvn archetype:generate -DgroupId=edu.eci.cvds -DartifactId=patterns -DarchetypeArtifactId=maven-archetype-quickstart

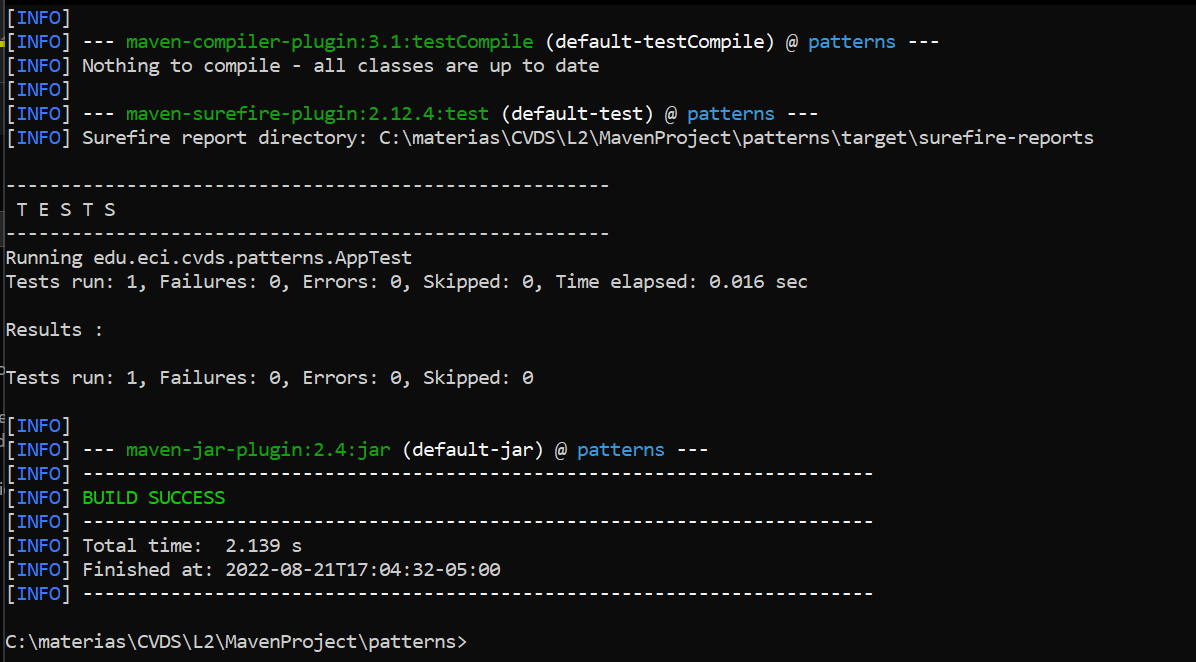
Comando tree:



AJUSTAR ALGUNAS CONFIGURACIONES EN EL PROYECTO



COMPILAR Y EJECUTAR



mvn package lo que hace es compilar el proyecto y generar un archivo .jar (en este caso) en el directorio target.

mvn clean: Limpia el proyecto borrando el directorio de destino.

mvn compiler:compile: Compila las clases de java del proyecto

mvn compiler:testCompile: Corre los archivos de prueba del proyecto

mvn install: Compila el proyecto e instala los archivos del proyecto en el repositorio local

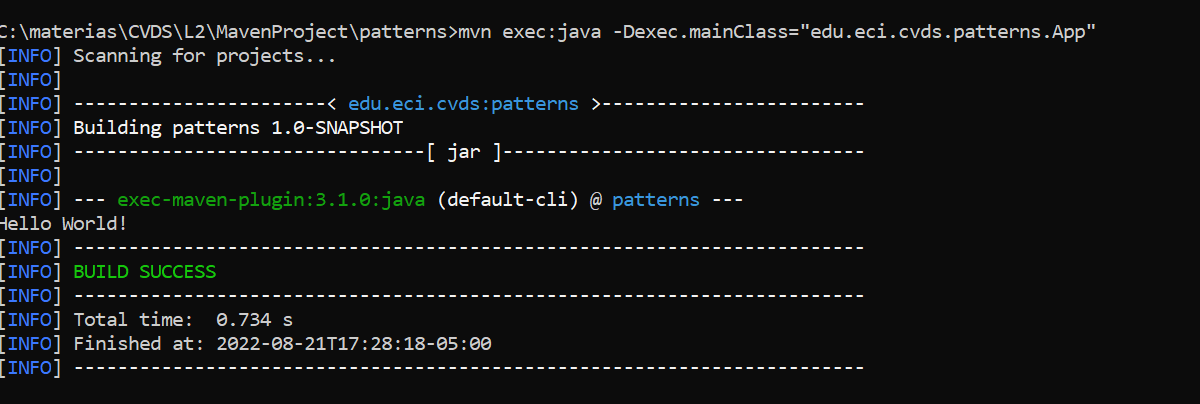
mvn deploy: Despliega el artefacto al repositorio remoto

mvn validate: Valida que todo esté correcto y la información necesaria esté disponible

Para ejecutar un proyecto de Maven de Java se debe usar el siguiente comando:

mvn exec:java -Dexec.mainClass="path.to.class"

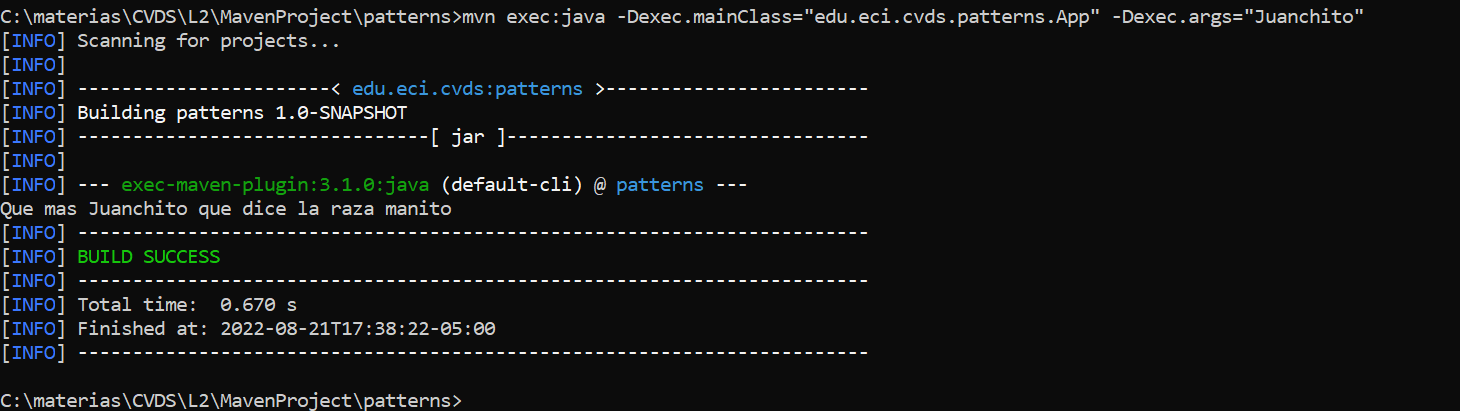
donde path.to.class es el path hasta la clase principal. El resultado fue el siguiente:



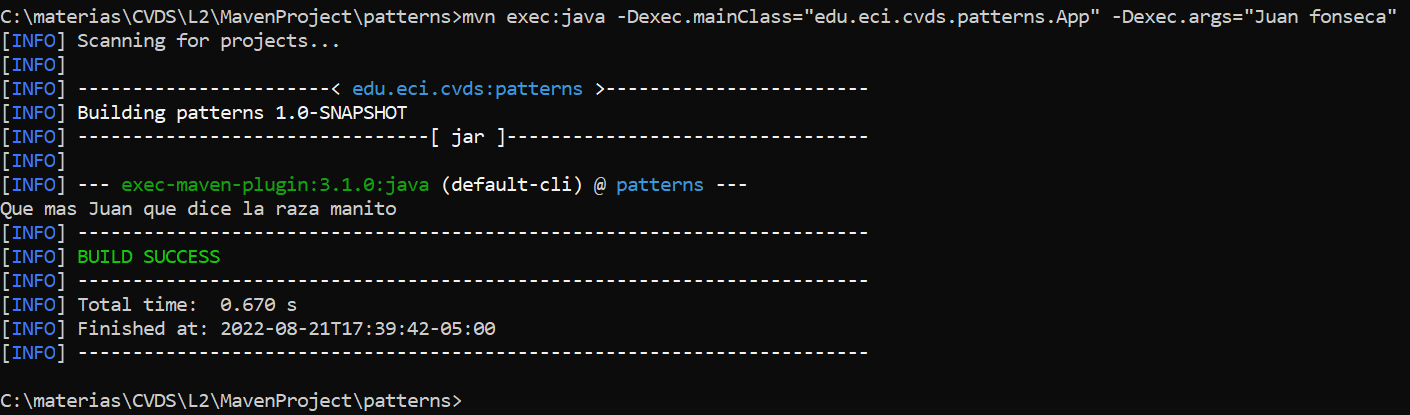
Ahí se evidencia que la salida es “Hello World!”.

Para enviar parámetros solo hay que agregar al comando anterior lo siguiente: -Dexec.args="arg".

Se modificó el saludo para que fuera personalizado y la salida fue la siguiente:



Se envió como parámetro “juan fonseca”, e imprimió lo siguiente:



Esto sucede porque para java cada espacio indica que hay otro argumento más, por ende el primer parámetro lo tomó como “juan” y no como “juan fonseca”.

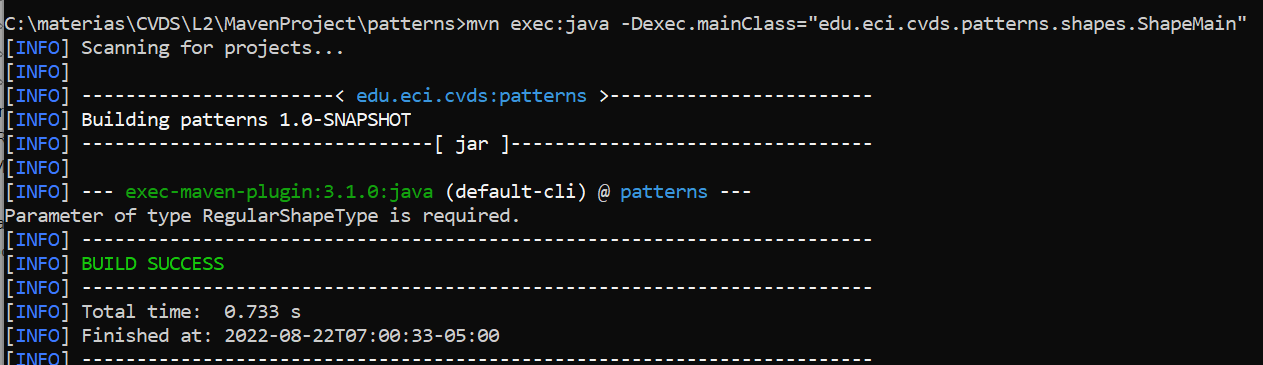
Para enviar un parámetro compuesto se debe de poner entre comillas simples como se aprecia a continuación.

Texto

Descripción generada automáticamente

Ejecución de ShapeMain:

* Sin parámetros:



* ‘qwerty’:

Texto

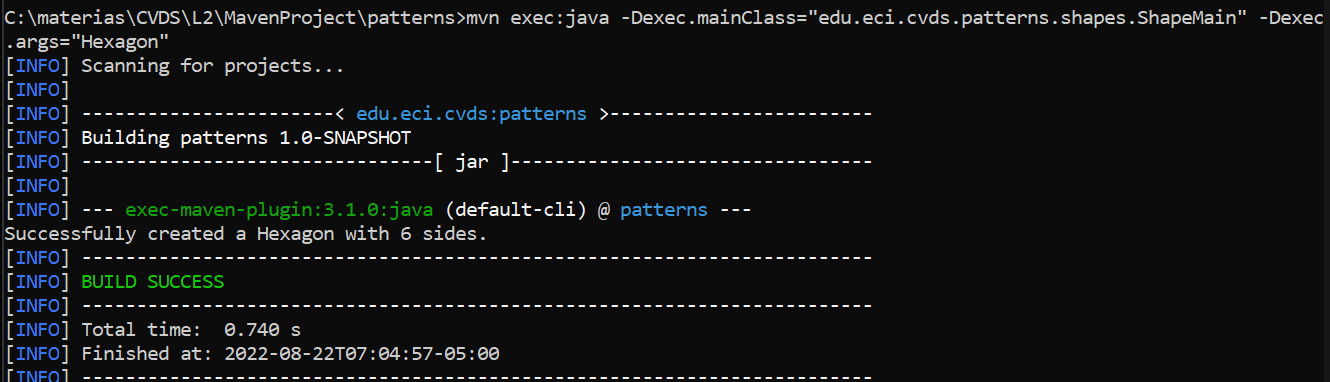
Descripción generada automáticamente

* ‘pentagon’:

Texto

Descripción generada automáticamente

* ‘Hexagon’:



De todas las ejecuciones la única que ejecutó satisfactoriamente fue con el parámetro ‘Hexagon’. Esto sucede porque esa palabra está en el enum de RegularShapeType.